|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 《数据库系统概论》实验报告 | | | | |
| 实验十 事务处理 | 姓名 | 刘爱兵 | 日期 | 2020.12.17 |
| 实验内容与完成情况：  **一、实验目的和要求**  了解事务的ACID属性，掌握事务和批处理的工作原理。  **二、实验任务**  四、实验内容：  1、设计并执行事务1：将学生“刘晨”选修2号课程的成绩改为99分。  BEGIN TRANSACTION  go  use 学生选课  go  update SC set Grade=99  where Sno in (select Sno from Student where Sname ='刘晨')  and Cno in (select Cno from Course where Cno='2')  go  commit  go    2、设计并执行事务2：将课程“数据结构”的课程号与“数据库”的课程号互换。  begin transaction  go  use 学生选课  go  declare @str1 int,@str2 int  select @str1 = Cno from Course where Cname = '数据结构'  select @str2 = Cno from Course where Cname = '数据库'  select Cno = @str2 from Course where Cname = '数据结构'  select Cno = @str1 from Course where Cname = '数据库'  go  commit  go    3、设计并执行事务3：要求学生“李勇”打算选修“操作系统”课程，根据规定，此门课程选修的人数最多为30人，该生是否可以选修此门课程，给出结果提示。  declare @renshu int  select @renshu = count(Distinct Sno)  from Course,Sc  where Cname = '操作系统'  if @renshu>30  print '人数超过'  else  print '人数未超过';    4、定义一个简单的事务，将sc表中所有课程成绩提高10%，只有全部成绩都更新成功，才提交整个事务。  BEGIN TRANSACTION  go  use 学生选课  go  update SC set Grade=Grade\*1.1  from SC  commit  go    设置事务隔离级别repeatable read.  假设希望检索student表中所有学生信息。首先在一个查询窗口中选择数据。为了防止其他事务更新正在读的数据，使用可重复读隔离。  use 学生选课  set transaction isolation level repeatable read  begin transaction  select \* from Student  update Student  set Sage=21  where Sno='200215121'    这时，查询会等待，不更新数据，因为共享锁会保持，以防止其他事务更新数据。  注意：若单击工具栏上的“取消执行查询”按钮，取消查询窗口2中的查询，而执行语句：  insert into Student VALUES ('201215126','abc','女',20,'IS')    将会在Student表中插入一个新行。（即使正处于可重复读隔离级别，这个语句也会被执行。这是因为，可重复读会锁定数据，以阻止数据的更新，但insert语句的插入是可以的，可以用SELECT \* FROM Student语句查看。）  5、创建存储过程insert\_student，可以通过该存储过程将学生的信息插入到表中，并能将所有学生的平均年龄返回给用户。（要求使用事务）  create procedure insert\_student  @sno char(10)=NULL, @sname char(20)=NULL,@sex char(2)=NULL,@age int=NULL, @avg\_age int output  AS  --对输入的参数进行检测，防止非空列输入空值  if @sno is NULL or @sname is NULL  --如果在非空列输入空值，返回提示信息  begin  print '必须输入学号，姓名和系部！'  print '请重新输入！'  return  end  else  --使用事务  begin tran  begin  insert into student(sno,sname,Ssex,Sage)  values(@sno,@sname,@sex,@age)  --检测插入是否成功  if @@error!=0  begin  rollback tran  print '插入数据失败，请重新输入！'  return  end  else  print '插入数据成功，学生姓名：'+@sname  end  commit tran  select @avg\_age=avg(Sage) from student  select @avg\_age as 平均年龄    6、执行存储过程insert\_student，将学生的信息（‘200205128’，李红，‘女’，24）插入到表student中, 并将所有学生的平均年龄返回给用户。  declare @avgage int  exec insert\_student '200205128','李红','女',24,@avgage    7、对SC表进行插入记录的工作，遇到错误时回滚到插入数据前的状态。  CREATE TRIGGER trig\_insert  ON SC--在SC表中创建trig\_insert触发器  AFTER INSERT--insert为触发事件，after则为触发的时机  AS  IF NOT EXISTS(SELECT \* FROM Course,inserted where Course.Cno=inserted.Cno)  --向SC表插入数据时，检查插入数据的课程号是否存在于Course表中  begin  print'插入的课程号不在课程表中'--出错提示  rollback  END    8、使用更新锁阻止其他用户对student表进行修改，但可以查询。如：  BEGIN TRAN  SAVE TRAN AAA  SELECT \* FROM Student WITH(UPDLOCK)  ROLLBACK TRAN AAA  COMMIT TRAN    9、使用排它锁阻止其他用户对course表进行访问。如：  BEGIN TRAN  SAVE TRAN BBB  SELECT \* FROM Course WITH(tablockx xlock)  ROLLBACK TRAN BBB  COMMIT TRAN    10、请设计一个银行业务数据库bankdb，其中，帐户信息表(bank)存放帐户的信息，交易信息表(transInfo)存放每次的交易信息。试用事务解决银行转帐问题：  思考题：1、上例并未考虑帐户余额不足情形，请完善之。  2、实验分别转帐1000和800时的转帐情况实现的过程。  3、实验结果分析。  CREATE TABLE BANK (  customername varchar(20), cardID varchar(10) , currentMoney float(4) )    CREATE TABLE TRANSINFO (  transDate datetime , cardID VARCHAR(50),  transType varchar(20) , check ( transType='支取' or transType='存入'), transMoney float(4), )    use bankdb  INSERT INTO BANK VALUES('张三','1001 0001','1000.0000') INSERT INTO BANK VALUES( '李四','1001 10002','1.0000')  USE BANKDB  INSERT INTO TRANSINFO VALUES('2005-10-11 11:30:46.623','1001 0001','支取','200.0000') | | | | |